Estimer la corrélation à la volée entre flux massifs est possible avec très peu de mémoire

Emmanuelle Anceaume, Yann Busnel

Estimer la corrélation à la volée entre flux massifs est possible avec très peu de mémoire

Emmanuelle Anceaume[1], - Yann Busnel[2]

[1] *IRISA & CNRS, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France*
[2] *Crest (Ensai) & LINA (Université de Nantes), campus de Ker Lann, rue Blaise Pascal, BP 37203, 35172 Bruz Cedex*
This work was partially funded by the French ANR project SocioPlug (ANR-13-INFR-0003), and by the DeSceNt project granted by the Labex CominLabs excellence laboratory (ANR-10-LABX-07-01).

L'analyse à la volée de flux massifs potentiellement infinis est fondamental dans un grand nombre d'applications de monitoring pour calculer des statistiques, détecter des tendances ou prédire des comportements déviants. En particulier, détecter la corrélation entre flux distribués semble être un bon indicateur de la présence d'attaques dans un réseau. Dans cet article, nous présentons une nouvelle métrique permettant d'évaluer la corrélation entre des flux distribués. Nous présentons un algorithme étonnamment simple et très peu couteux en espace permettant d'estimer à la volée la corrélation entre des flux. Nous dérivons des bornes sur la qualité de l'évaluation, et validons notre approche sur des traces réelles.

Keywords: Flux de données, métrique de corrélation, algorithme distribué d'approximation probabiliste.

1 Introduction

Performance of many complex monitoring applications, including Internet monitoring applications, data mining, sensors networks, network intrusion/anomalies detection applications, depend on the detection of correlated events[JY04] . To cope with the complexity and the amount of raw data, current network management tools analyze their input streams in isolation. The point is that, in all these monitoring applications, data streams arrive at nodes in a very high rate and may contain up to several billions of data items per day. Thus computing statistics with traditional methods is unpractical due to constraints on both available processing capacity and memory. Two main approaches exist to monitor in real time massive data streams. The first one consists in regularly sampling the input streams so that only a limited amount of data items is locally kept. Accuracy of this computation fully depends on the volume of data items that has been sampled and their order in the stream. Furthermore, an adversary may easily take advantage of the sampling policy to hide its attacks among data items that are not sampled, or in a way that prevents its "malicious" data items from being correlated. In contrast, the streaming approach consists in scanning each piece of data of the input stream on the fly, and in locally keeping only compact synopses or sketches that contain the most important information about these data, deriving some data streams statistics with guaranteed error bounds without making any assumptions on the order in which data items are received at nodes. On the other hand, very few works have tackled the distributed streaming model, also called the functional monitoring problem [CMY08], which combines features of both the streaming model and communication complexity models, where each node receives an input data stream, performs some local computation, and communicates only with a coordinator who wishes to continuously compute or estimate a given function of the union of all the input streams. The challenging issue in this model is for the coordinator to compute the given function by minimizing the number of communicated bits [CMY08]

In this paper, we go a step further by studying the dispersion matrix of distributed streams. Specifically, we propose a novel correlation metric, called the sketch codeviation, that allows to quantify in real time how observed data items, between distributed and massive streams, change together, and in which proportion. We give upper and lower bounds on the quality of this approximated metric with respect to the codeviation.

We then provide a distributed algorithm that additively approximates the codeviation among n data streams $\sigma_1,\ldots,\sigma_n$ by using $O((1/\varepsilon)\log(1/\delta)(\log N+\log m))$ bits of space for each of the n nodes, where N is the domain size from which items values are drawn, and m is the largest size of these data streams. We guarantee that for any $0<\delta<1$, the maximal error of our estimation is bounded by $\varepsilon m/N$, by sending $O(n(1+(1/\varepsilon)\log(m/2)\log(1/\delta)))$ bits among the whole network.

2 Model and Metrics

Model We consider a set of n nodes $S_1,\ldots,S_n$ such that each node S_i receives a large sequence σ_i of data items or symbols. We assume that streams $\sigma_1,\ldots,\sigma_n$ do not necessarily have the same size and support (which are unknown). Items arrive regularly and quickly, and due to memory constraints, need to be processed sequentially and in an online manner. Each data item j is drawn from the universe $\Omega=\{1,2,\ldots,N\}$, where N is very large. Nodes cannot communicate among each other. On the other hand, there exists a specific node, called the *coordinator* in the following, with which each node may communicate. We assume that communication is instantaneous. A natural approach to study a data stream σ_i of length m_i is to model it as a fingerprint vector (or item frequency vector) over the universe Ω, given by $X_i=(x_1,x_2,\ldots,x_N)$ where x_j represents the number of occurrences of data item j in σ_i. Note that $0\leq x_j\leq m_i$.

Codeviation We propose a metric over fingerprint vectors of items, which is inspired from the classical covariance metric in statistics. Such a metric allows us to qualify the dependance or correlation between two quantities by comparing their variations. As will be shown in Section 4, this metric captures shifts in the network-wide traffic behavior when a DDoS attack is active. The codeviation between any two fingerprint vectors $X=(x_1,x_2,\ldots,x_N)$, and $Y=(y_1,y_2,\ldots,y_N)$ is the real number denoted $\mathrm{cod}(X,Y)$ defined by

$$\mathrm{cod}(X,Y)=\frac{1}{N}\sum_{i\in\Omega}(x_i-\overline{x})(y_i-\overline{y})=\frac{1}{N}\sum_{i\in\Omega}x_iy_i-\overline{x}\,\overline{y}\quad\text{ where }\overline{x}=\frac{1}{N}\sum_{i\in\Omega}x_i\text{ and }\overline{y}=\frac{1}{N}\sum_{i\in\Omega}y_i. \tag{1}$$

3 Distributed Codeviation Approximation Algorithm

We propose a statistic tool, named the sketch codeviation, which allows to approximate the codeviation between any two data streams using compact synopses or sketches. This paper presents the main results of our study. For space limitation reasons, proofs are presented in the companion paper [AB14].

Définition 3.1 (Sketch codeviation) *Let X and Y be any two fingerprint vectors of items, such that $X=(x_1,\ldots,x_N)$ and $Y=(y_1,\ldots,y_N)$. Given a precision parameter k, we define the sketch codeviation between X and Y as*

$$\widehat{\mathrm{cod}_k}(X,Y)=\min_{\rho\in\mathcal{P}_k(\Omega)}\mathrm{cod}\left(\widehat{X}_\rho,\widehat{Y}_\rho\right)=\min_{\rho\in\mathcal{P}_k(\Omega)}\left(\frac{1}{N}\sum_{a\in\rho}\widehat{X}_\rho(a)\widehat{Y}_\rho(a)-\left(\frac{1}{N}\sum_{a\in\rho}\widehat{X}_\rho(a)\right)\left(\frac{1}{N}\sum_{a\in\rho}\widehat{Y}_\rho(a)\right)\right)$$

where $\forall a\in\rho,\widehat{X}_\rho(a)=\sum_{i\in a}x_i$, and $\mathcal{P}_k(\Omega)$ is a k-cell partition of Ω, i.e.*, the set of all the partitions of the set Ω into exactly k nonempty and mutually disjoint sets (or cells).*

Proposition 3.2 *Let $X=(x_1,\ldots,x_N)$, and $Y=(y_1,\ldots,y_N)$ be any two fingerprint vectors. The sketch codeviation is a function of the codeviation. We have $\widehat{\mathrm{cod}_N}(X,Y)=\mathrm{cod}(X,Y)$ and*

$$\widehat{\mathrm{cod}_k}(X,Y)=\mathrm{cod}(X,Y)+\mathcal{E}_k(X,Y)\quad\textit{where}\quad\mathcal{E}_k(X,Y)=\min_{\rho\in\mathcal{P}_k(\Omega)}\frac{1}{N}\sum_{a\in\rho}\sum_{i\in a}\sum_{j\in a\setminus\{i\}}x_iy_j.$$

We have shown in Theorem 4 of the companion paper [AB14] that the sketch codeviation matches exactly the codeviation if $k\geq|\,\mathrm{supp}(X)\cap\mathrm{supp}(Y)\,|+\mathbf{1}_{\mathrm{supp}(X)\setminus\mathrm{supp}(Y)}+\mathbf{1}_{\mathrm{supp}(Y)\setminus\mathrm{supp}(X)}$, where $\mathrm{supp}(X)$, respectively $\mathrm{supp}(Y)$, represents the support of distribution X, respectively Y (*i.e.*, the set of items in Ω that have a non null frequency $x_i\neq 0$, respectively $y_i\neq 0$, for $1\leq i\leq N$), and notation $\mathbf{1}_A$ denotes the indicator function, which is equal to 1 if the set A is not empty and 0 otherwise. We have also characterized the upper bound of the overestimation factor, *i.e.*, the error made with respect to the codeviation, when k is strictly less than this bound.

Theorème 3.3 (Upper bound of $\mathcal{E}_k(X,Y)$) *Let $k \geq 1$ be the precision parameter of the sketch codeviation. For any two fingerprint vectors $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$, let $\mathcal{E}_k$ be the maximum value of the overestimation factor $\mathcal{E}_k(X,Y)$. Then, the following relation holds.*

$$\mathcal{E}_k = \max_{X \in \mathcal{X}, Y \in \mathcal{Y}} \mathcal{E}_k(X,Y) = \begin{cases} \frac{m_X m_Y}{N} & \text{if } k = 1, \\ \frac{m_X m_Y}{N}\left(\frac{1}{k} - \frac{1}{N}\right) & \text{if } k > 1. \end{cases}$$

Distributed approximation algorithm We compute the codeviation between a set of n distributed data streams, so that the number of bits communicated between the n sites and the coordinator is minimized. This amounts for the coordinator to compute an approximation of the codeviation matrix Σ, which is the dispersion matrix of the n data streams. Specifically, let $\mathbb{X} = \{X_1, X_2, \ldots, X_n\}$ be the set of fingerprint vectors $X_1, \ldots, X_n$ describing respectively the streams $\sigma_1, \ldots, \sigma_n$. We have

$$\widehat{\Sigma} = \left[\widehat{\mathrm{cod}}(X_i, X_j)\right]_{1 \leq i \leq n, 1 \leq j \leq n}.$$

We propose a one-pass algorithm that computes the sketch codeviation between any two large input streams. By definition of the Sketch codeviation metric, we need to generate all the possible k-cell partitions. The number of these partitions follows the Stirling numbers of the second kind, which is equal to $S(N,k) = \frac{1}{k!}\sum_{j=0}^{k}(-1)^{k-j}\binom{k}{j}j^N$. Therefore, $S(N,k)$ grows exponentially with N, which is unreasonable in term of space complexity as N is supposed to be very large. Actually generating $t = \lceil \log(1/\delta) \rceil$ random k-cell partitions, where δ is the probability of error of our randomized algorithm, is sufficient to guarantee good overall performance of the sketch codeviation metric [AB14]. Our approximation algorithm uses an elegant data structured inspired by [CM05] to generate those t partitions on the fly. The algorithm proceeds in rounds until all the data streams have been read in their entirety. In the following, we denote by $\sigma_i^{(r)}$ the substream of σ_i received by S_i during the round r, and by d_r the number of data items in this substream. During each round, each site S_i computes a single sketch C_{σ_i} of the received data stream σ_i, until receiving d_r items (See [AB14] for details on the data structure C_{σ_i}). Then node S_i sends $C_{\sigma_i^{(r)}}$ to the coordinator, keeps a copy of $C_{\sigma_i^{(r)}}$, and starts a new round $r+1$. Upon receipt of $C_{\sigma_i^{(r)}}$ from any S_i, the coordinator asks all the $n-1$ other nodes S_j to send their own sketch $C_{\sigma_j^{(r)}}$. Once the coordinator has received all $C_{\sigma_i^{(r)}}$, the coordinator locally updates the n sketches such as $C_{\sigma_i} \leftarrow C_{\sigma_i} + C_{\sigma_i^{(r)}}$ and updates the sketch codeviation matrix $\widehat{\Sigma} = \left[\widehat{\mathrm{cod}}(X_i, X_j)\right]_{1 \leq i \leq n, 1 \leq j \leq n}$ on every couple of sketches, such that the element in position i, j represents the sketch codeviation between streams σ_i and σ_j. As the codeviation is symmetric, the codeviation matrix is a symmetric matrix, and thus only the upper-triangle and the diagonal need to be computed. In order to make this algorithm self-adaptive to the unknown lengths of the streams, d_r is more or less doubled at each round. Note that during round r, S_i regularly computes $\mathrm{cod}\left(\sigma_i^{(r-1)}, \sigma_i^{(r)}\right)$ to detect whether significant variations in the stream have occurred before having received d_r items. This allows to inform the coordinator as quickly as possible that some attack might be undergoing, by sending its current sketch $C_{\sigma_i^{(r)}}$ earlier than scheduled.

Theorème 3.4 *The approximated codeviation matrix $\widehat{\Sigma}$ returned by the distributed sketch codeviation algorithm satisfies $\widehat{\Sigma} \geq \Sigma$ and*

$$\mathbb{P}\left\{\left|\widehat{\Sigma} - \Sigma\right| \geq \frac{\varepsilon}{N} \max_{i,j \in [n]} \left(\|X_i\|_1 \|X_j\|_1 - \|X_i X_j\|_1\right)\right\} \leq \delta.$$

using $O((1/\varepsilon)\log(1/\delta)(\log N + \log m))$ bits of space for each n nodes, and $O(n \log m (1/\varepsilon \log(1/\delta) + n))$ bits of space for the coordinator, where m is the maximum size among all the streams, i.e., $m = \max_{i \in [n]} \|X_i\|_1$. *Moreover, the distributed sketch codeviation algorithm gives an approximation of matrix Σ by sending $O(rn(1 + (1/\varepsilon)\log(m/2)\log(1/\delta)))$ bits, where r is the number of the last round.*

4 Performance Evaluation

We have implemented the distributed sketch codeviation algorithm and have conducted a series of experiments on different types of streams and for different parameters settings. We have fed our algorithm with both real-world data sets and synthetic traces. Due to space constraints, we only describe one of these experiments, which is mainly representative of our algorithm's accuracy. The interested reader is invited to consult [AB14] for a complete overview of our evaluation, which clearly shows that our distributed algorithm is capable of efficiently and accurately quantifying how observed data streams change together and in which proportion whatever the shape of the input streams.

Figure 1 shows how efficiently our approximation distributed algorithm detects different scenarii of attacks in real time. Specifically, we compute at each round of the distributed protocol, the distance between the codeviance matrix Σ constructed from the streams under investigation and the mean of covariance matrices $\mathbb{E}(\Sigma_N)$. This distance has been proposed in [JY04]. Specifically, given two square matrices M and M' of size n, consider the distance $\|M-M'\| = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}(M_{i,j}-M'_{i,j})^2}$. We evaluate at each round r, the variable d_r defined by $d_r = \|\Sigma_r - \mathbb{E}(\Sigma_N)\|$.

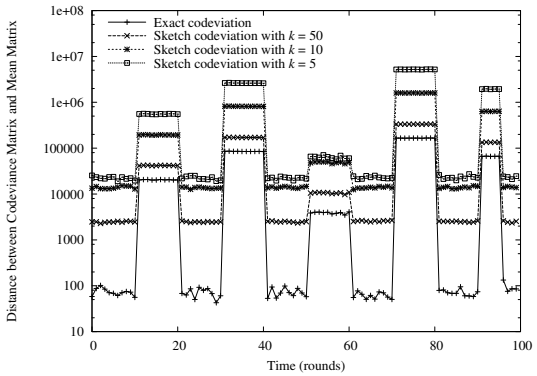

FIGURE 1: Distance between the codeviation matrix and the mean of the past ones when all the 10 synthetic traces follow different distributions as a function of the rounds of the protocol, with $\delta = 10^{-5}$.

Based on this distance, we have fed our distributed algorithm with different patterns of traffic. In Figure 1, distance is depicted, as a function of time, when the codeviance is exactly computed and when it is estimated with our distributed algorithm with different values of k. What can be seen is that, albeit there are up to two orders of magnitude between the exact codeviance matrix and the estimated one, the shape of the codeviance variations are for most of them similar. Different attack scenarii are simulated. From round 0 to 10, all the 10 synthetic traces follow the same nominal distribution (*e.g.*, a Poisson distribution). Then from round 10 to 20 a targeted attack is launched by flooding a single node (*i.e.*, one among the ten traces follows a Zipfian distribution with $\alpha = 4$). This gives rise to a drastic and abrupt increase of the distance. As can be shown, the estimated covariance exactly follows the exact one, which is a very good result. Then after coming back to a "normal" traffic, half of the traces are replaced by Zipfian ones (from round 30 to 40), representing a flooding attack toward a group of nodes. As for the previous attack, the covariance matrices are highly impacted by this attack. From round 50 to 60, traces follow a Zipfian distribution with $\alpha = 1$ which represents unbalanced network traffic but should not be completely representative of attacks. On the other hand, in the fourth and fifth attack periods, all the traces follow a Zipfian distribution with different values of $\alpha \geq 2$, which clearly shows a flooding attack toward a group of targeted nodes. The main lesson drawn from these results is the good performance of our distributed algorithm whatever the pattern of the attack.

Références

[AB14] Emmanuelle Anceaume and Yann Busnel. Deviation estimation between distributed data streams. In *Proc. of the 10th European Dependable Computing Conference (EDCC)*, 2014.

[CM05] G. Cormode and S. Muthukrishnan. An improved data stream summary : the count-min sketch and its applications. *Journal of Algorithms*, 55(1) :58–75, 2005.

[CMY08] G. Cormode, S. Muthukrishnan, and K. Yi. Algorithms for distributed functional monitoring. In *Proc. of the 19th Annual ACM-SIAM Symposium On Discrete Algorithms (SODA)*, 2008.

[JY04] Shuyuan Jin and D.S. Yeung. A covariance analysis model for ddos attack detection. In *4th IEEE International Conference on Communications (ICC '04)*, volume 4, pages 1882–1886, 2004.